Resources @CNAF

You can monitor the usage of the muoncoll resources here.

More information about the use of Tier1 resources can be found in this guide.

Preliminary setup

To access to the grid resources @CNAF you need an account on the CNAF gateway: *bastion.cnaf.infn.it*. Follow this guide for the instructions on how to get an account at CNAF.

Then use the general purpose UI: **ui-tier1.cr.cnaf.infn.it**

```
ssh <accountName>@bastion.cnaf.infn.it

ssh <accountName>@ui-tier1.cr.cnaf.infn.it
```

To use grid resources you need a personal certificate. Instructions can be found here.

User certificate and key are usually saved in the files $HOME/.globus/usercert.pem and $HOME/.globus/userkey.pem resp.

You need also to register to the VO muoncoll: follow this link.

Proxy

To generate a proxy:

```
voms-proxy-init --voms muoncoll.infn.it --vomslife 24:00 --valid 24:00
```

This creates a proxy valid for 24 hours with a VO extension. You can check your proxy with the command:

```
-bash-4.2$ voms-proxy-info -all

subject : /DC=org/DC=terena/DC=tcs/C=IT/O=Istituto Nazionale di Fisica Nucleare/CN=Alessio Gianelle gianelle@infn.it/CN=1473510485
issuer : /DC=org/DC=terena/DC=tcs/C=IT/O=Istituto Nazionale di Fisica Nucleare/CN=Alessio Gianelle gianelle@infn.it
identity : /DC=org/DC=terena/DC=tcs/C=IT/O=Istituto Nazionale di Fisica Nucleare/CN=Alessio Gianelle gianelle@infn.it
type : RFC3820 compliant impersonation proxy
strength : 1024
path : /tmp/x509up_u62503
timeleft : 23:59:55
key usage : Digital Signature, Key Encipherment
=== VO muoncoll.infn.it extension information ===
VO : muoncoll.infn.it
subject : /DC=org/DC=terena/DC=tcs/C=IT/O=Istituto Nazionale di Fisica Nucleare/CN=Alessio Gianelle gianelle@infn.it
issuer : /DC=org/DC=terena/DC=tcs/C=IT/L=Frascati/O=Istituto Nazionale di Fisica Nucleare/OU=Istituto Nazionale di Fisica Nucleare/CN=voms2.cnaf.infn.it
attribute : /muoncoll.infn.it/Role=NULL/Capability=NULL
timeleft : 23:59:55
uri : voms2.cnaf.infn.it:15022
```

If the job is longer than 24 hours, it is going to be aborted for proxy expiration time. To extend the lifetime of the proxy you need to store a proxy credential in a dedicated store.

On Tier1@CNAF the myproxy store is myproxy.cnaf.infn.it. With this command you should store a credential on the myproxy server, it will ask first your certificate password, then you have to set a proxy password that should be inserted in the submission file:

```
myproxy-init --proxy_lifetime 24 --cred_lifetime 720 --voms muoncoll.infn.it --pshost myproxy.cnaf.infn.it --dn_as_username --credname proxyCred --local_proxy
```

The maximum lifetime of long-lived proxies on a MyProxy server is one week (168 hours), and it can be prolonged with 24 hour steps, to achieve this:

```
myproxy-logon --pshost myproxy.cnaf.infn.it --dn_as_username --credname proxyCred --proxy_lifetime 24
```

Last, you have to modify your condor submit file adding these lines:

```
use_x509userproxy = true
MyProxyHost = myproxy.cnaf.infn.it:7512
MyProxyPassword = "put your proxy password"
MyProxyCredentialName = proxyCred
MyProxyRefreshThreshold = 600                    The time (in second) before the expiration of a proxy that the proxy should be refreshed
MyProxyNewProxyLifetime = 1440                   The new lifetime (in minutes) of the proxy after it is refreshed
```

The previous described method sometimes failed. So inside the VO muoncoll.infn.it there is the possibility to ask for a proxy with a duration of 48 hours. This means that you can submit job that can stay on the grid's queues (i.e. IDLE + RUN time) for at most 2 days. Remember to add

this option to your submit file: *`delegate_job_GSI_credentials_lifetime = 0`*

Condor jobs

A simple guide can be found here.

Submit job

First of all, set GSI as the authentication method:

```
export _condor_SEC_CLIENT_AUTHENTICATION_METHODS=GSI
```

There are six computing elements for grid submission: `ce01-htc.cr.cnaf.infn.it`, `ce02-htc.cr.cnaf.infn.it`, `ce03-htc.cr.cnaf.infn.it`, `ce04-htc.cr.cnaf.infn.it`, `ce05-htc.cr.cnaf.infn.it`, `ce06-htc.cr.cnaf.infn.it`.

To submit to the second CE you have to use the command:

```
condor_submit -pool ce02-htc.cr.cnaf.infn.it:9619 -remote ce02-htc.cr.cnaf.infn.it -spool test.sub
```

where test.sub is the submit file which represents the job.

Query job

To check the job status of a single job use:

```
condor_q -pool ce02-htc.cr.cnaf.infn.it:9619 -name ce02-htc.cr.cnaf.infn.it <condorID>

condor_q -pool ce02-htc.cr.cnaf.infn.it:9619 -name ce02-htc.cr.cnaf.infn.it -l <condorID>

condor_q -pool ce02-htc.cr.cnaf.infn.it:9619 -name ce02-htc.cr.cnaf.infn.it -better-analyze  <condorID>
```

You can also get the status of all your jobs, your matched user can be discovered with this command:
`condor_q -pool ce02-htc.cr.cnaf.infn.it:9619 -name ce02-htc.cr.cnaf.infn.it -l  <condorID>  | grep Owner`

```
condor_q -pool ce02-htc.cr.cnaf.infn.it:9619 -name ce02-htc.cr.cnaf.infn.it  <matchedUser>
```

Get output

When the job is finished retrieve the output with the command:

```
condor_transfer_data -pool ce02-htc.cr.cnaf.infn.it:9619 -name ce02-htc.cr.cnaf.infn.it  <condorID>
```

Remove job

If something went wrong remove the job with:

```
condor_rm -pool ce02-htc.cr.cnaf.infn.it:9619 -name ce02-htc.cr.cnaf.infn.it   <condorID>
```

Status

Check the status of the available resources using:

```
condor_status -pool ce02-htc.cr.cnaf.infn.it:9619 -state -avail
```

Use singularity image

The muoncoll software is released also through docker images or singularity images stored on the cvmfs area: **/cvmfs/muoncoll.infn.it/sw/singularity/**

The following submit file:

- requires 4GB of memory;
- sets log, output and error files as unique appending the CondorID;
- asks to transfer back the simulation and reconstruction files: sim.out and reco.out.

test.sub

```
use_x509userproxy = true

delegate_job_GSI_credentials_lifetime = 0

+owner = undefined

request_memory = 4GB

executable = test.sh

transfer_input_files = job.sh

log = test_$(ClusterId).$(ProcId).log

output = outfile_$(ClusterId).$(ProcId).txt

error = errors_$(ClusterId).$(ProcId).txt

transfer_output_files = sim.out, reco.out

should_transfer_files = Yes

when_to_transfer_output = ON_EXIT

queue
```

The executable file:

- creates the *jobHome* directory and copy inside the job's script: *job.sh;*
- uses the singularity command to exec the job's script (NB: mount the jobHome directory as $HOME);
- uses gfal util to transfer the output of the simulation to the SE.

test.sh

```
#!/bin/bash

mkdir jobHome
mv job.sh jobHjobHomeome

singularity exec -B jobHome:$HOME  /cvmfs/muoncoll.infn.it/sw/singularity/MuonColl_v02-05-MC.sif /bin/bash $HOME/job.sh

### If you want you can copy the output to the SE

gfal-copy jobHome/MuonCutil/SoftCheck/muonGun_sim.slcio srm://storm-fe-archive.cr.cnaf.infn.it:8444/muoncoll/

mv jobHome/sim.out .

mv jobHome/reco.out .

### clean dir

rm -rf jobHome
```

The job's script is executed inside the singularity container:

- sources the init file to set the environment;
- clones the git repository with the example scripts;
- cd into the SoftCheck directory inside the repository;
- runs the simulation and the reconstruction example, saving the output in the $HOME directory.

job.sh

```
#!/bin/bash

source /opt/ilcsoft/muonc/init_ilcsoft.sh

git clone https://github.com/MuonColliderSoft/MuonCutil.git

cd MuonCutil/SoftCheck

GEO="/opt/ilcsoft/muonc/detector-simulation/geometries/MuColl_v1/MuColl_v1.xml"

ddsim --compactFile ${GEO} --steeringFile sim_steer.py &> $HOME/sim.out

Marlin --InitDD4hep_mod4.DD4hepXMLFile=${GEO} reco_steer.xml &> $HOME/reco.out
```